Exploring Large Language Models' Ability to Describe Entity-Relationship Schema-Based Conceptual Data Models

information

MDPI

Article

Exploring Large Language Models' Ability to Describe Entity-Relationship Schema-Based Conceptual Data Models

Andrea Avignone *,†,‡, **Alessia Tierno** †,‡, **Alessandro Fiori** †,‡ **and Silvia Chiusano** †,‡

Department of Control and Computer Engineering, Politecnico di Torino, 10129 Turin, Italy; s303464@studenti.polito.it (A.T.); alessandro.fiori@polito.it (A.F.); silvia.chiusano@polito.it (S.C.)
* Correspondence: andrea.avignone@polito.it
† Current address: Corso Duca degli Abruzzi, 24, 10129 Torino, Italy.
‡ These authors contributed equally to this work.

Abstract: In the field of databases, Large Language Models (LLMs) have recently been studied for generating SQL queries from textual descriptions, while their use for conceptual or logical data modeling remains less explored. The conceptual design of relational databases commonly relies on the entity-relationship (ER) data model, where translation rules enable mapping an ER schema into corresponding relational tables with their constraints. Our study investigates the capability of LLMs to describe in natural language a database conceptual data model based on the ER schema. Whether for documentation, onboarding, or communication with non-technical stakeholders, LLMs can significantly improve the process of explaining the ER schema by generating accurate descriptions about how the components interact as well as the represented information. To guide the LLM with challenging constructs, specific hints are defined to provide an enriched ER schema. Different LLMs have been explored (ChatGPT 3.5 and 4, Llama2, Gemini, Mistral 7B) and different metrics (F1 score, ROUGE, perplexity) are used to assess the quality of the generated descriptions and compare the different LLMs.

Keywords: relational database; large language models; database design; entity-relationship

Academic Editor: Katsuhide Fujita

Received: 10 March 2025
Revised: 11 April 2025
Accepted: 29 April 2025
Published: 29 April 2025

Citation: Avignone, A.; Tierno, A.; Fiori, A.; Chiusano, S. Exploring Large Language Models' Ability to Describe Entity-Relationship Schema-Based Conceptual Data Models. *Information* **2025**, *16*, 368. https://doi.org/10.3390/info16050368

1. Introduction

Databases are the backbone of modern information systems, enabling efficient storage, retrieval and management of data. A key phase for relational databases is the conceptual modeling, where the entity-relationship (ER) model is widely used to represent the data structure and constraints before translating them into a logical relational schema by means of translation rules (i.e., the generation of tables using SQL). An ER schema represents information through three key constructs: (i) **entities**, classes of objects with independent existence; (ii) **relationships**, logical associations between entities; and (iii) **attributes**, properties of both entities and relationships. A proper conceptual model is crucial because it serves as the foundation for translating real-world requirements into a consistent database structure. By accurately representing the elements and constraints within a system, a well-designed conceptual model ensures data integrity, consistency, and scalability throughout the database lifecycle [1–4].

The ER data model plays an important role not only in the initial design of databases but also in their documentation and long-term maintenance. Although the ER data model is a simple and intuitive representation, textual documentation is an indispensable support to integrate the ER schema with descriptions of the information represented in the schema itself. In fact, a clear human-readable documentation of an ER schema—concerning the

informative content of each construct—is essential for effective communication among database designers, developers, and business stakeholders.

In recent years, Large Language Models (LLMs) have shown remarkable capabilities in natural language understanding and generation, finding applications in various fields, including software development and database systems [5–7]. Much of the existing research on LLMs in the database domain has focused on automating the query generation process to interact with databases using natural language, the definition of the final logical schema according to user requirements, and the textual explanation of tables, relationships, and constraints in an existing database [8,9].

LLMs also have the potential to serve as a powerful tool for automating the generation of natural language descriptions of ER schemas, improving the comprehensibility of database structures and information represented in the schema itself. They can provide detailed explanations of the schema, including the relationships between entities and their properties. As an example, Figure 1 shows the portion of an initial ER schema for a database about a library loan system and the informative description corresponding to the ER obtained by employing an LLM. The text demonstrates how an LLM can understand the properties beyond the two entities (BOOK and AUTHOR) and the binary relationship linking them (AUTHOR_OF), and transform a structured schema into an accurate description. The obtained text is intuitively helpful for *communication* with expert and non-expert users without the need to interpret (complex) ER schemas. By facilitating the deeper understanding of the database design, it serves as *documentation* useful for knowledge transfer, reference, and future updates. It also supports the *validation* of the schema to ensure that the requirements of the system are met.

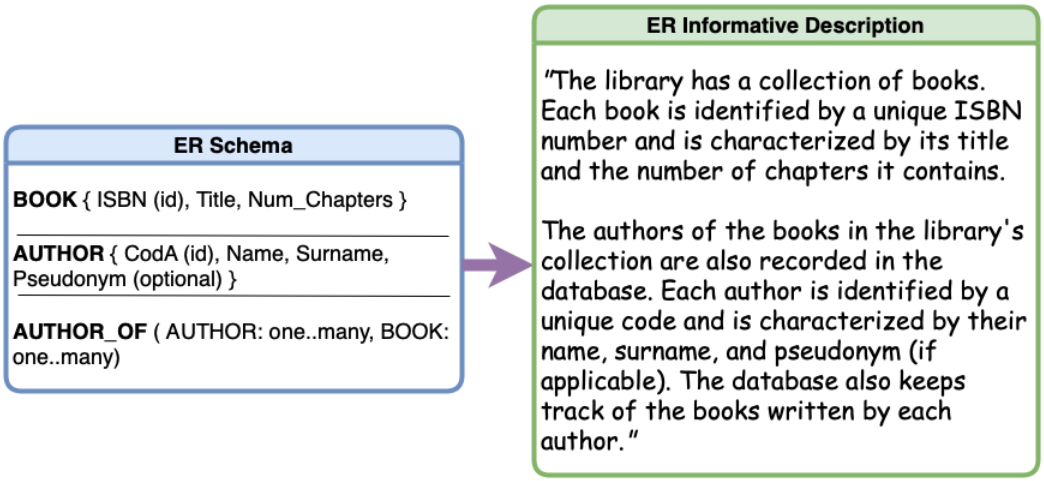


Figure 1. An illustrative example of generating textual descriptions with a Large Language Model starting from the ER schema for a library system.

This paper explores the capabilities of LLMs in generating accurate, complete, and easily understandable textual descriptions of conceptual ER schemas. Our study involves testing various LLMs to identify the limitations of each model and assess their ability to *understand* and *describe* the main ER constructs with examples covering different levels of difficulty. Additionally, we explore strategies for enriching the conceptual schema with contextual-driven hints to guide LLMs in producing more precise and readable descriptions. The main contributions of this paper are as follows:

- A comprehensive investigation into the performance of various LLMs (ChatGPT 3.5 and 4, LLama2, Gemini, Mistral 7B) in generating accurate and readable textual descriptions of ER schemas with varying complexity.

- The exploration of effective enrichment strategies and techniques to improve the prompt and guiding LLMs to generate more precise descriptions of specific ER constructs.
- The design of a framework for preprocessing the input ER schemas and postprocessing the generated textual descriptions, along with a comparison of LLM performance using evaluation metrics such as F1 score, ROUGE, and perplexity.
- The release of a publicly available repository with test cases of ER schemas and the corresponding database requirements, supporting future research in this area.

Our investigation is guided by the following research questions (RQs):

RQ1: *What is the LLM's ability to recognize an ER construct and what are the challenges?*

Provide a quantitative assessment of LLMs' ability to recognize and interpret the core constructs of an ER schema. We define a construct as properly "recognized" when the LLM provides an accurate description, identifying the type of construct and thoroughly detailing its properties (i.e., the characteristics of the information it represents). Our hypothesis is that an LLM that fails to correctly recognize a construct will struggle to generate a proper textual description of the construct or the whole ER schema. The goal is to qualitatively identify common issues, such as incompleteness or lack of understandability, that may arise during this process. These issues could be related to specific characteristics of the input schemas or inherent limitations of LLMs themselves. For example, we investigate which types of constructs are inaccurately or incompletely described, whether constructs are misrepresented or substituted, and the impact of these issues on the generated descriptions.

RQ2: *What is the quality of the textual description generated by an LLM from a given ER schema?*

Evaluate the quality of the textual descriptions generated by LLMs when provided with an ER schema. We use quantitative measures (i.e., F1 score, ROUGE, and perplexity) to assess effectiveness and applicability in practical contexts.

RQ3: *How can LLMs be guided or supported to generate high-quality textual descriptions?*

Identify strategies for improving the quality of the textual descriptions generated by LLMs. We categorize these strategies into two main areas: (1) enriching the ER model with keywords or hints to guide the LLM in generating a more accurate description, and (2) adjusting LLM prompts to better align with the task of generating detailed and coherent textual descriptions. These actions seek to optimize the interaction between the ER model and the LLM to enhance the quality of the generated descriptions.

2. Related Work

Generating descriptive text from structured data represents an important task with applications across multiple domains. Researchers have explored various approaches to this challenge, including fine-tuning pre-trained models [10–12] and reconstructing table structures to better capture their inherent information [13]. These methods have demonstrated effectiveness in transforming structured information into natural language descriptions that preserve semantic content while presenting it in an accessible format.

The emergence of Large Language Models (LLMs) has significantly impacted various aspects of software engineering. Recent systematic literature reviews [7] have highlighted the broad applicability of these models across the software development lifecycle, from requirement engineering to code generation and testing. Research by [14] has examined both the opportunities and potential risks of applying LLMs to software engineering tasks, noting their potential to enhance productivity while raising concerns about generalizability and education. Similarly, ref. [15] has contributed to our understanding of how LLMs perform across different software engineering challenges, providing insights into their capabilities and limitations. In the context of requirement engineering specifically, ref. [16]

has conducted a comprehensive review of ChatGPT applications, studying how an LLM can properly integrate in the software requirements practices to improve the process.

Building on these broader applications of LLMs in software engineering, the interaction between these models and databases has attracted significant research interest. A substantial portion of this research has concentrated on Text-to-SQL translation [8,17,18], where natural language queries are automatically converted into executable SQL statements; and SQL-to-Text to explain SQL code [19]. These systems aim to make database interaction more intuitive for non-technical users by removing the need for SQL expertise. Text-to-SQL approaches have been successfully applied in diverse domains [20–22], with recent advancements including schema-aware systems designed to mitigate data imbalance challenges [23]. Such schema-aware approaches incorporate database structural information to improve translation accuracy and handle the variability in database designs.

Beyond Text-to-SQL translation, researchers explored LLMs for conceptual model understanding and generation. Recent studies have investigated methods to automatically convert system requirements expressed in natural language into UML diagrams using various LLM architectures [24,25]. This direction of research demonstrates the potential of LLMs to understand abstract conceptual relationships and represent them in structured formats. In [26], the authors used Retrieval Augmented Generation Large Language Models (RAG-based LLMs) to support teaching activities related to conceptual modeling.

While these applications have significantly advanced database usability and automation, comparatively less attention has been given to the process of leveraging LLMs for generating comprehensive textual descriptions of entity-relationship (ER) conceptual schemas. ER schemas serve as foundational representations of database structure, capturing entities, relationships, and constraints in a format that is often challenging for non-technical stakeholders to interpret. This research gap presents an opportunity for novel contributions that could enhance database comprehension, facilitate knowledge transfer, and support documentation efforts across organizations.

3. Materials and Methods

Figure 2 illustrates our proposed framework for transforming an entity-relationship (ER) schema into a descriptive human-readable text using different LLMs. The process consists of different steps:

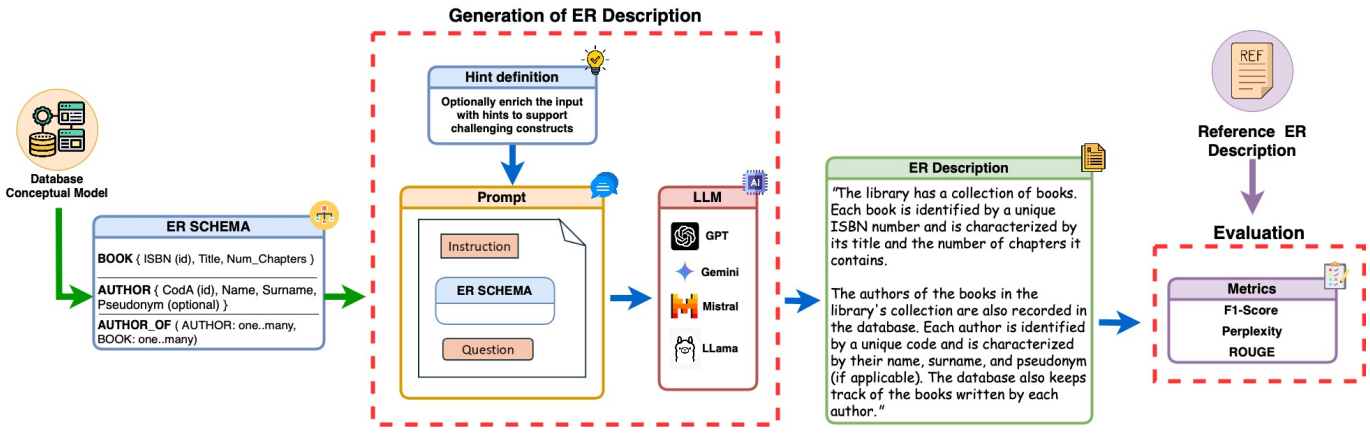


Figure 2. Proposed framework overview for transforming ER conceptual schemas into descriptive text using LLMs.

1. **Data Preparation.** The system takes as input the conceptual ER schema of a specific database. In our experiments, we designed a dataset collecting ER schemas and the corresponding textual description written by human experts (Sections 3.1 and 3.2).

2. **LLM prompting and hint definition.** The processed schema is converted into a structured prompt and fed into the LLM, which generates the descriptive text output (Section 3.3.1). Optionally, an enrichment step adds hints for complex constructs to guide the model in generating more accurate descriptions (Section 3.3.2).
3. **Evaluation.** The generated text is assessed using multiple metrics, including F1 score, perplexity, and ROUGE, to measure accuracy and fluency (Section 3.4).

3.1. ER Schema Formalism

To provide structured textual input to the LLMs, we defined a custom yet precise textual representation of the ER schemas. Since there is no widely adopted or standardized textual syntax for describing ER schemas in natural language or structured text, we adopted a compact convention that is both human-readable and machine-parsable. Our formalism was designed to preserve the semantics of ER constructs—such as entities, attributes, relationships, cardinalities, and generalizations—while enabling consistent interpretation by language models. The following patterns describe the syntax we adopted for encoding the core components of ER models.

Entity Definition.

```
EntityName { Attr_1 (id), Attr_2, Attr_3 (optional), Attr_4 (multi),
    Attr_5 { SubAttr_1, SubAttr_2 } }
```

Notes:

- `(id)` denotes a unique identifier and it can characterize more than one attribute.
- `(optional)` marks an attribute as non-mandatory.
- `(multi)` denotes a multivalued attribute.
- Composite attributes are written as `Attr { SubAttr1, SubAttr2 }`, where `SubAttr1` and `SubAttr2` are sub-attributes of `Attr`.

Relationship Definition.

```
RelName (EntityA: card, EntityB: card, ...) {RelAttr_1, RelAttr_2 (
    optional), RelAttr3, ...}
```

Notes:

- `card` refers to the entity participation in the relationship in the `min..max` notation, with `min` ∈ (`zero`, `one`), `max` ∈ (`one`, `many`).
- The curly braces `{RelAttr_1, RelAttr_2 (optional), RelAttr_3, ...}` are omitted if no relationship attribute exists.
- Entity in the relationship with an external identifier participates with cardinality `card=one..one` and includes the `external id` term.

Generalizations.

```
ParentEntity <= { ChildEntity1, ChildEntity2, ... } (partial/total,
    exclusive/overlapping)
```

Notes:

- `partial` — not every occurrence of the parent must belong to a child.
- `total` — every occurrence of the parent entity must be in a child.
- `exclusive` — child entities are disjoint.
- `overlapping` — occurrences of the parent entity may belong to multiple child entities.

Example Snippet.

As an example, a snippet of an ER schema described using the above textual notation is provided here. This model describes the structure of a database designed to manage an academic institution's staff, students, and courses.

```
STAFF { SID (id), Name, Surname, OfficePhone, MobilePhone (optional) }
FACULTY { Position }
ADMINISTRATIVE { JobType }
STAFF <= {ADMINISTRATIVE, FACULTY} (partial, exclusive)
COURSE { CID (id), Name, Credits, Degree }
TEACHING (FACULTY: zero..many, COURSE: one..many ) { TotalHours }
LECTURE { Date (external id), Subject, Room }
SCHEDULING (LECTURE: one..one external id, COURSE: one..many)
```

The `STAFF` entity represents university personnel, uniquely identified by the attribute `SID` and characterized by the attributes: first name, last name, office phone number, and optionally the mobile phone number. Since there are various roles, including administrative staff and teachers, the `STAFF` entity serves as the parent in a (partial and exclusive) generalization, with child entities `FACULTY` and `ADMINISTRATIVE`. Faculty members are characterized by their current position, while administrative staff are described by job type.

The `COURSE` entity models the courses offered by the institution, identified by a unique `CID`, and includes attributes such as name, number of credits, and degree. The `TEACHING` relationship models faculty participation in courses. It connects entities `FACULTY` and `COURSE` with cardinalities `zero..many` and `one..many`, respectively; and includes the attribute `TotalHours`, indicating the number of hours a faculty member contributes to a course.

The `LECTURE` entity models individual course lectures. A lecture is identified by the date when the lecture is scheduled (attribute `Date`) and by the course for which the lecture is offered (by means of an external identifier). The `SCHEDULING` relationship, which links entities `LECTURE` and `COURSE`, supplies the external identifier for `LECTURE`.

3.2. Dataset

Experimental validation was conducted on 13 test cases, each corresponding to the definition of an entity-relationship (ER) model for different application contexts (Table 1). The characteristics of these datasets, in terms of ER constructs present in the schema, are summarized in Table 2. A complete description of each test case, including the ER schema according to the formalism reported in Section 3.1, and a reference description of the schema created by a human expert, is available in a publicly accessible GitHub repository (https://github.com/AndreaAvignone/ER2Text.git, accessed on 10 March 2025).

Table 1. Application domains of the ER models used in the test cases.

Test Case	Application Domain
Test 1	Multimedia platform
Test 2	Library loan
Test 3	Boat rental
Test 4	Radiological examinations
Test 5	Server
Test 6	Cluster of servers
Test 7	Information about territorial control by officers
Test 8	Online newspaper publisher
Test 9	Kitchen appliance stores

Table 1. *Cont.*

Test Case	Application Domain
Test 10	Museum restoration
Test 11	Free-floating e-scooter sharing rental company
Test 12	Online services as movie rental, online music, and video games
Test 13	Airline company

Table 2. Summary of ER schema characteristics across test cases.

ER Construct	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Test 11	Test 12	Test 13
Entity													
Entity	7	5	5	9	8	8	4	7	8	8	5	5	7
Relationship													
Binary Relationship	2	3	3	4	3	3	1	3	5	3	2	1	5
Ternary Relationship	0	0	0	1	1	1	1	1	0	1	1	1	0
Generalization													
Generalization	2	1	1	1	1	1	0	1	1	1	0	1	1
Identifier													
Simple Identifier	2	3	3	5	4	4	2	4	4	5	5	3	4
Composite Identifier	0	0	0	1	1	1	1	0	0	1	0	0	0
External Identifier	1	1	1	2	1	1	1	1	1	0	0	1	2
Attribute													
Simple Attribute	10	10	7	16	16	16	10	12	16	8	9	11	11
Composite Attribute	0	0	0	0	0	0	1	0	1	0	1	0	1
Multivalued Attribute	1	1	1	3	1	1	1	1	0	1	1	0	1
Optional Attribute	1	2	1	3	1	1	2	1	1	3	0	0	0

The ER schemas used in the study vary in complexity across two difficulty levels. High-difficulty models include more complex constructs, such as ternary relationships, foreign keys, generalizations. In contrast, medium-difficulty schemas contain fewer constructs and often lack ternary relationships. These test cases were designed to evaluate the performance of LLMs when presented with training examples of mixed difficulty levels.

3.3. Generation of ER Informative Description

In this section, we report how the LLM prompt is generated and we discuss some key examples to enrich the ER schema for specific constructs.

3.3.1. LLM Prompting and Inference

Adapting the prompt based on the task to be performed is necessary and after a careful and extensive study of various techniques [27,28], it was concluded that only a subset of them are applicable to this research: Zero-Shot, N-shot (specifically One-Shot and Few-Shot), and Chain-of-Thought:

- Zero-Shot prompting [29,30] is a technique where a model is given a task without any prior examples or specific training for that task. Therefore, the model relies on its pre-existing knowledge and understanding to generate responses based on the prompt alone.
- N-Shot prompting [27,31] involves the addition of one or more task-specific examples, followed by a natural language description of the task. In this way, the model has sufficient context to generalize and perform the task effectively.
- Chain-of-Thought [32,33] is the prompting technique that encourages LLMs to explain the reasoning that led them to their answer. Usually, the prompt includes a few-shot

examples followed by the explanation of the reasoning that led to generating the result. This approach guides the model to use a similar process.

We conducted an exploratory analysis of the prompts above in the targeted problem. The Zero-Shot approach is not usable to obtain sufficient responses. Both the Few-Shot and Chain-of-Thought techniques give optimal and comparable results. However, the Chain-of-Thought requires more effort to create the prompts and more space to store them, as well as the need for postprocessing to remove the reasoning part. For these motivations, the Few-Shot proves to be the best choice. In subsequent analyses, we focus only on this prompt.

3.3.2. Hint definition for ER schema enrichment

This step aims to explore the ability of LLMs to understand and describe ER constructs and their properties. It also focuses on defining possible suggestions to guide the LLM in properly describing specific constructs where the LLM struggles to correctly interpret the model. We consider the ER construct correctly identified when the LLM produces an informative description that satisfies the following properties: (i) It properly describes the *construct properties*; (ii) It accurately explains the *information* that the construct models.

For our explorative analysis, we used a collection of simple reference ER schemas, each one including a given ER construct, such as an entity, a binary or ternary relationship, or a generalization. These constructs may also include attributes (simple or composite). In the case of an entity, identifiers can be simple, composite, or external. For each reference ER schema, the LLM is asked to generate a proper description of the ER construct.

Our analysis pointed out that the LLMs considered are capable of correctly identifying and properly describing the main constructs of the ER model. These constructs include entity, identifiers (simple and composite), attributes (simple, optional, composite, and multivalued), and relationships. For some more complex constructs, the system provides a less accurate and not entirely correct description of the construct and its properties. If the LLM does not recognize the construct correctly, we enhance the original description to guide the LLM in the identification process without the need of tuning the model parameters to meet all the different examples and constructs.

Hints. We propose to introduce *hints* in the original ER schema to influence the LLM's understanding of the ER construct and providing a more accurate description. The hints are defined by explicitly labeling each specific construct in which the LLM struggles to properly interpret the model. Enriching the initial specification during automatic preprocessing with some hints allowed this problem to be overcome, guiding the LLM in understanding the ER schema and generating the expected output. We found hints to be necessary for the following ER constructs: (i) *ternary relationships* and (ii) *generalizations*. Hints to be included for these constructs are described in Table 3. In Section 3.3.3, we discuss some example cases in which the hints are introduced.

Table 3. Defined hints to support LLMs with complex ER constructs.

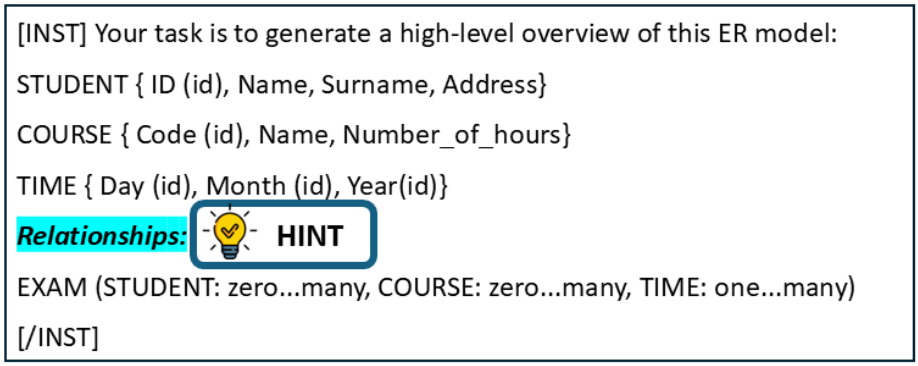
Construct	Hint
Ternary Relationships	Introduce term *Relationships* before ternary construct.
Generalizations	Introduce term *Generalizations* before generalization construct.

3.3.3. Example Cases

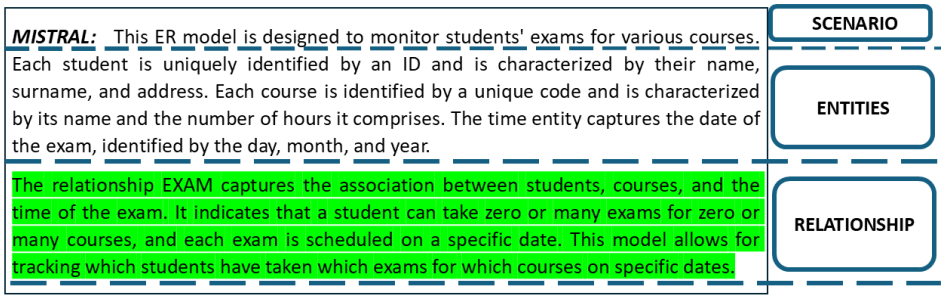
Entities and Relationships. Entities and relationships are the key components in the ER model. Entities present classes of objects with an autonomous existence, while relationships represent logical links between two or more entities. Entities and binary relationships (linking two entities) are correctly recognized by the different LLMs, providing an accurate

description without the need for additional hints. In contrast, LLMs can benefit from the introduction of a specific hint when a ternary relationship is involved to provide a more accurate description of the information represented. A ternary relationship (linking three entities) is a more complex construct, as it models concepts expressed by the jointly co-occurrences of three involved entities.

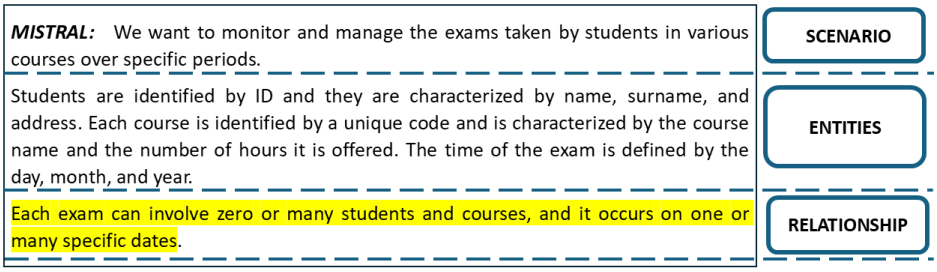
Figure 3a shows an example of an ER schema representing the exams taken by students at different times. This concept is modeled using a ternary relationship, called EXAM, linking the entities STUDENT, COURSE, and TIME.



(**a**)



(**b**)



(**c**)

Figure 3. Ternary construct (Mistral): (**a**) Prompt. (**b**) Textual description obtained with a hint. (**c**) Textual description obtained without a hint.

We introduced the hint `Relationship` to support the LLM in recognizing the correct ER construct represented by EXAM. This hint suggests that EXAM functions as a relationship rather than an entity, establishing connections between three distinct entities: STUDENT, COURSE, and TIME. By incorporating this terminology, we guide the model to understand that EXAM represents a ternary relationship linking these three conceptual entities together.

Based on Figure 3b, the LLM correctly identifies and introduces the target scenario, the involved entities, and the relationship. For each of the three entities, the LLM provides a short description of the *information* modeled in terms of the concept represented by the construct (*entity name*) and the related properties that characterize (*attributes*) and uniquely

identify (*identifiers*) the entity occurrences. For example, for the STUDENT entity, it reports that each student is uniquely identified by an "ID" (*identifiers*) and characterized by name, surname, and address (*attributes*). For the TIME entity, it reports that this entity captures the date of the exam, identified by the day, month, and year (*composite identifier*).

Then, a short description of the *information* modeled by the EXAM relationship is provided. It reports that EXAM captures the association between the students, the courses, and the time of the exam, allowing us to track exams taken by students on specific dates. It also reports that each student can optionally participate in the relationship, but can take exams for more courses, and that exams are scheduled on specific dates (in accordance with the cardinality of the EXAM relationship specified in the ER schema).

Figure 3c shows the description provided by the LLM when the hint is omitted from the ER schema. It can be seen that, in this case, the description of the information represented by the EXAM relationship is much less accurate and complete, and succinctly represented in a single sentence. In addition, it is not explicitly stated that the examination represents a relationship of the ER schema.

ER descriptions in Figure 3b,c were generated by Mistral and they are reported as an example; a similar solution (with some variations in the notation) is obtained using other LLMs. Without hints, the description of the relationship tends to be less accurate for older LLMs (LLama and ChatGPT3.5), with the ternary relationship decomposed into three binary relationships.

Generalizations. Generalizations represent logical links between an entity E and one or more entities E1, which participate in the generalization with a different *role*. Entity E has the role of *parent entity* and entities E1...En have the role of *child entities*. Entity E is more general, in the sense that it comprises them as a particular case. This leads to the *inheritance property*, stating that a child entity is characterized by the same properties (attributes, identifiers, relations, and other generalizations) of the parent entity, along with possibly other distinctive properties. The *covering properties* specify if every occurrence of the parent entity is also an occurrence of one of the child entities (total) or not (partial), and if every occurrence of the parent entity is at most an occurrence of one of the child entities (disjoint) or not (overlapped).

An LLM that provides an accurate description of the generalization construct must be able to correctly identify and specify the key elements: (i) *construct name* (generalization); (ii) *role of involved entities* (parent and child entity); (iii) *inheritance property*; (iv) *covering properties* (partial or total, overlapped or exclusive); and (v) readable description of the information represented by the construct.

Figure 4 shows an example of a (partial and exclusive) generalization construct where the ATHLETE entity is a generalization of SKIER and SAILOR entities. We introduced the `Generalization` hint to support the LLM in correctly recognizing and describing the construct. It is introduced before listing the construct characteristics to clearly suggest the type of construct.

Based on Figure 4b, the LLM correctly identifies the target scenario, the construct, and the modeled information. For the ATHLETE entity, it reports that each athlete is identified by a unique CardNumber (*identifier*) and characterized by name and surname (*attributes*). For SKIER and SAILOR entities, it reports that they are among the occurrences of athlete (i.e., *child entities*) that are represented by additional properties, such as speciality and role, respectively. Moreover, the description explicitly indicates that the construct is a generalization and briefly explains that it is partial and exclusive—not all athletes are necessarily skiers or sailors, and each athlete can be either a skier or a sailor, but not both.

Figure 4c shows the description provided by the LLM when the hint is omitted. The critical point is that the ER construct is incorrectly referred to as "relationship" holding

"between athletes and their specific roles (skiers or sailor)". This description seems to suggest that the concepts of athlete, skier, and sailor are perceived as three separate and independent concepts, whereas the ER model indicates the presence of only one concept (athlete) of which the other two are specializations.

ER descriptions in Figure 4b,c were generated by Mistral and they are reported as an example; a similar solution (with some variations in the notation) is obtained using other LLMs.

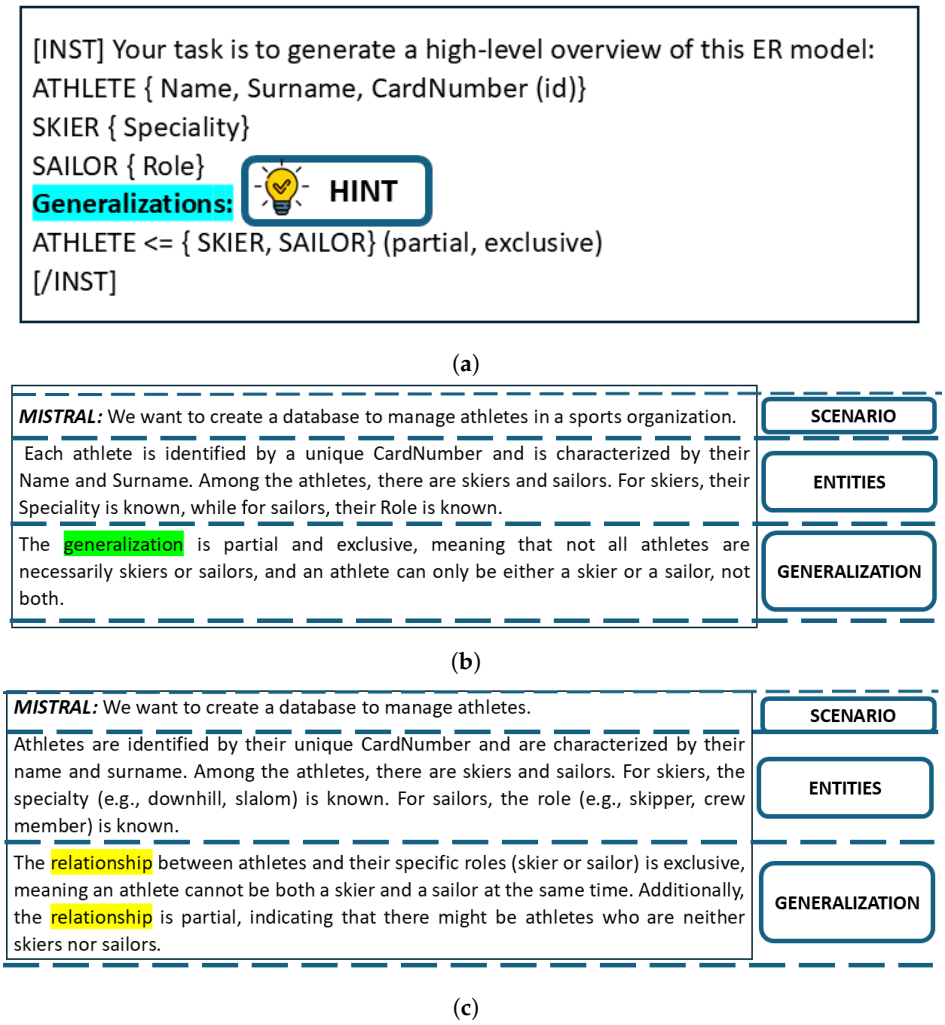


Figure 4. Generalization construct (Mistral): (**a**) Prompt. (**b**) Textual description obtained with a hint. (**c**) Textual description obtained without a hint.

3.4. ER Informative Description Evaluation

To assess the quality of the descriptive text generated from an ER schema using LLMs, several evaluation metrics can be used, each focusing on different aspects of the text. Since the text is descriptive and informational (rather than creative), metrics that evaluate clarity, accuracy, and coverage are more relevant.

3.4.1. F1 Score

The *F1 Score* is used to measure the overlap between the generated textual description and a reference description, balancing both *precision* (how much of the generated text is correct) and *recall* (how much of the reference text is captured). In our evaluation, we compute the F1 score by comparing the LLM-generated textual description with the

input ER schema. This metric helps us determine how well the LLM captures the correct constructs present in the schema. The F1 score is defined as

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{1}$$

where the following definitions apply:

- **Precision** is the proportion of correctly described constructs among all the ones produced by the LLM.
- **Recall** is the proportion of correctly described constructs among all the ones in the reference description.

3.4.2. Perplexity

Perplexity (PPL) is a measure of how well a language model predicts a given sequence of text [34,35]. A lower perplexity value indicates higher fluency and coherence in the generated description. It is defined as

$$PPL = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log P(w_i|w_{1:i-1})\right) \tag{2}$$

where the following denotions apply:

- $P(w_i|w_{1:i-1})$ is the probability assigned to the i-th token in the generated text given all previous tokens.
- N is the total number of tokens in the generated description.

Since textual descriptions of ER schemas must be both readable and well structured, perplexity allows us to evaluate how well LLMs generate text that resembles human-written descriptions. A model that assigns lower probabilities to the correct next word will have higher perplexity, meaning that the model is making less accurate predictions and producing more "surprising" text. It follows that the lower the perplexity, the better the model predictions.

The range of acceptable perplexity values depends on the context of the problem being analyzed. For natural language texts, perplexity is considered good if it is below 50. Other factors that influence perplexity include the model used to calculate it, the length of the sequences, and the complexity of the language involved. For these reasons, perplexity is most useful when used to compare texts generated by different models.

Perplexity is based purely on the model's probability predictions and does not directly measure whether the generated text makes sense in a real-world context or captures deeper meanings. In practice, while perplexity is a helpful starting point for model evaluation, it is often used alongside other metrics such as ROUGE (Section 3.4.3).

3.4.3. ROUGE Score

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a widely used metric for evaluating text generation tasks by measuring the overlap between the generated text and a reference text. In general, ROUGE measures how closely the generated text matches the reference text in terms of overlap of words, phrases, or sequences. ROUGE includes several variants, each focusing on different aspects of similarity:

- **ROUGE-N**: Measures n-gram overlap between the generated and reference texts. The most commonly used is **ROUGE-1** (unigram overlap), which evaluates word-level similarity, and **ROUGE-2** (bigram overlap), which assesses phrase-level similarity.
- **ROUGE-L**: Computes the longest common subsequence (LCS) between the generated and reference texts, capturing sentence-level structure.

The ROUGE score primarily relies on three key measures, precision, recall, and F1 score, each capturing different aspects of text similarity:

- Precision: The proportion of n-grams (e.g., unigrams, bigrams) in the generated text that also appear in the reference text. High precision indicates that the generated text contains mostly relevant information.
- Recall: The proportion of n-grams in the reference text that are correctly included in the generated text. High recall suggests that the generated description captures most of the key information.
- F1 score: The harmonic mean of precision and recall, providing a balanced evaluation when both aspects are important. It is particularly useful when the goal is to maximize both completeness and correctness of the generated text.

The ROUGE score is typically normalized between 0 and 1, where higher values indicate greater lexical overlap between the generated and reference texts. However, in descriptive and informative tasks, such as generating textual descriptions of ER schemas, exact lexical matches are not always necessary. A good description should capture the core concepts and relationships of the schema, even if the wording differs. ROUGE-1 is particularly suitable for our task as it quantifies the lexical similarity between the textual description generated by an LLM and the reference description of an ER schema, created by a human expert. Since the goal is to generate coherent and informative explanations of ER schemas, this metric provides insights about whether the essential terms and constructs are correctly conveyed.

4. Results

This section presents the results of our exploratory analysis comparing different LLMs for generating human-readable descriptions from ER conceptual schemas. We evaluate the models based on F1 score, perplexity, and ROUGE metrics.

4.1. F1 Score

The F1 score reveals satisfactory results, showing comparable performance for all LLMs in understanding the constructs of ER schemas, as shown in Table 4.

Table 4. F1 score comparison of various Large Language Models.

Model	Mean	Standard Deviation (SD)
ChatGPT-3.5	0.81	0.16
ChatGPT-4	**0.88**	0.09
Gemini	0.80	0.10
Llama 2	0.71	0.19
Mistral 7B	0.84	**0.07**

In general, ChatGPT-4 and Mistral 7B emerge as the most promising options in terms of F1 score. Both demonstrate superior performance with the highest F1 score (MEAN = 0.88 and MEAN = 0.84, respectively) and the lowest variability (SD=0.09 and SD = 0.07, respectively). This combination of high mean performance and low variability makes ChatGPT-4 and Mistral 7B particularly reliable for ER schema description tasks, with consistent performance across different test cases.

ChatGPT-3.5 and Gemini achieve a respectable F1 score (MEAN = 0.81 and MEAN = 0.80, respectively), although Gemini shows lower variability (SD = 0.10) and thus more consistent across the different samples. Llama 2 demonstrates the lowest performance among the tested models with an F1 score (MEAN = 0.71) and the highest variability

(SD = 0.19). This suggests that Llama 2 may be less suitable for ER schema description tasks where consistency is important.

4.2. Perplexity

Table 5 summarizes the results in terms of perplexity for the LLMs under study. Lower perplexity values indicate better language modeling capabilities, as the model assigns higher probability to the correct tokens. It quantifies how much the model generates more predictable text.

Table 5. Perplexity score comparison of various Large Language Models.

Model	Mean	Standard Deviation (SD)
ChatGPT-3.5	20.06	5.53
ChatGPT-4	16.95	3.70
Mistral 7B	13.01	5.44
Llama	**10.49**	**2.45**
Gemini	13.10	2.69

The perplexity comparison reveals interesting patterns in their predictive capabilities for ER schema description tasks. Since perplexity is considered acceptable for values below 50 for natural language texts, all the LLMs show satisfactory results.

Llama2 achieves the lowest average perplexity (MEAN = 10.49) and standard deviation (SD = 2.45), indicating superior next-token prediction capability and consistency in this context. Mistral 7B and Gemini follow closely in terms of perplexity (MEAN = 13.01 and MEAN = 13.10), but with Gemini showing lower variability for different ER schemas (SD = 2.69) than Mistral 7B (SD = 5.44). ChatGPT-4 exhibits moderately higher perplexity (MEAN = 16.95) compared to Llama2, Mistral 7B, and Gemini, but its variability (SD = 3.70) indicates relatively consistent performance across different schemas. ChatGPT-3.5 demonstrates the highest perplexity (MEAN = 20.06), with considerable variability (SD = 5.53).

These results present an interesting contrast to the F1 score measurements, particularly regarding Llama2, which showed the lowest F1 scores but the best perplexity metrics. This apparent contradiction suggests that lower perplexity alone may not translate directly to better overall performance in ER schema description tasks. The findings indicate that models optimized for next-token prediction (as measured by perplexity) may not necessarily excel at capturing the semantic accuracy required for high F1 scores in this domain. The descriptions provided by Llama 2 are very smooth and clear; however, the ER models described often deviate excessively from the provided ones, making the descriptions frequently unusable. These observations highlight the importance of using multiple evaluation metrics when assessing model performance for specialized tasks like ER schema description.

4.3. ROUGE Score

Table 6 presents the ROUGE scores for different LLMs used in our study. These scores evaluate the lexical overlap between the generated text and the reference schema descriptions by a human expert, providing insight into how well each model preserves the original content.

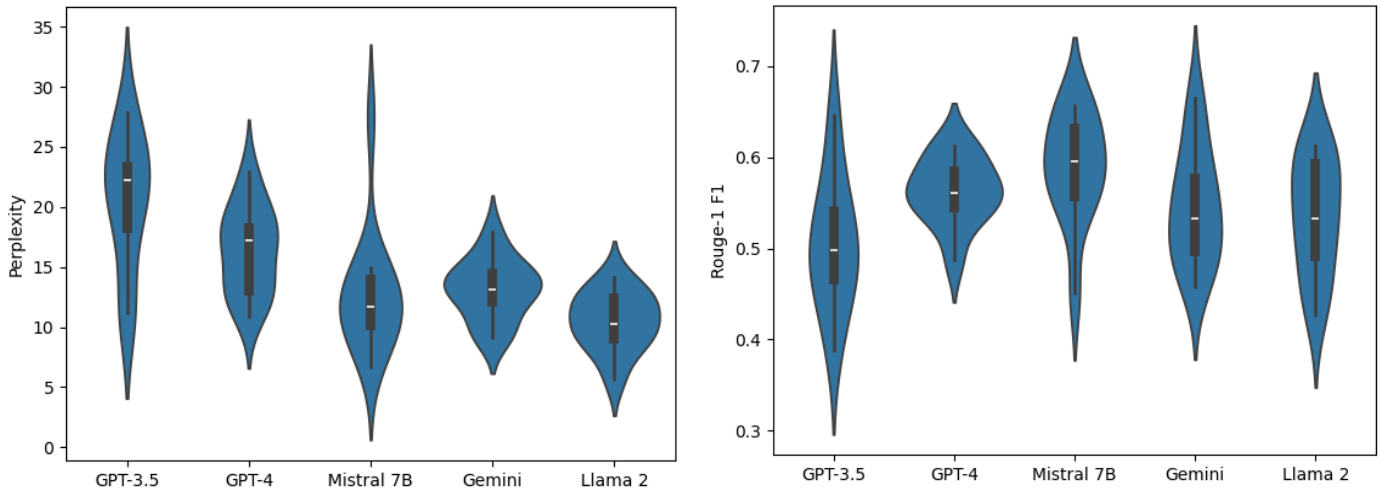
Mistral 7B demonstrates superior overall performance with the highest ROUGE-1 F1 (MEAN = 0.589) and precision (MEAN = 0.564), indicating strong unigram matching capabilities. This model also performs best on ROUGE-2 precision, recall, and F1, suggesting better ability to capture bigram sequences accurately. It similarly leads in ROUGE-L metrics, achieving the highest scores in precision (MEAN = 0.305) and F1 (MEAN = 0.320). Mistral 7B generates text that is lexically similar and consistent with the reference descriptions.

ChatGPT-4 follows maintaining a strong balance between precision and recall. ChatGPT-4 also demonstrates the lowest standard deviation in ROUGE-1 F1 (SD = 0.037), indicating the most consistent performance across different ER schema test cases. Other models, such as Gemini and Llama 2, also perform well but do not achieve the same level of consistency as Mistral 7B. ChatGPT-3.5 exhibits the highest ROUGE-1 recall (MEAN = 0.645), indicating its tendency to include relevant tokens from the reference text. However, its precision scores are notably lower considering the other ROUGE metrics, thus it may generate descriptions with additional tokens and a reduced level of lexical similarity with the reference descriptions.

Table 6. ROUGE score comparison of various LLMs. Higher values indicate better overlap between generated and reference texts.

		ROUGE-1			ROUGE-2			ROUGE-L		
Model		**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
GPT 3.5	MEAN	0.427	**0.645**	0.507	0.143	0.211	0.168	0.233	**0.353**	0.277
	SD	0.103	0.050	0.070	0.059	0.050	0.053	0.060	0.055	0.052
GPT 4	MEAN	0.516	0.634	0.563	0.187	0.227	0.203	0.278	0.341	0.303
	SD	0.074	0.059	0.037	0.047	0.035	0.038	0.055	0.060	0.046
Llama 2	MEAN	0.485	0.606	0.535	0.165	0.206	0.182	0.255	0.318	0.281
	SD	0.072	0.083	0.065	0.061	0.067	0.061	0.063	0.077	0.062
Mistral 7B	MEAN	**0.564**	0.625	**0.589**	**0.233**	**0.261**	**0.245**	**0.305**	0.341	**0.320**
	SD	0.075	0.082	0.060	0.050	0.067	0.053	0.060	0.083	0.065
Gemini	MEAN	0.496	0.633	0.546	0.188	0.239	0.207	0.274	**0.353**	0.303
	SD	0.085	0.103	0.064	0.057	0.064	0.057	0.053	0.082	0.056

To better stress the distribution of the models and compare the behavior with different metrics, in Figure 5, we report the violin plot for both perplexity (Figure 5a) and ROUGE-1 F1 score (Figure 5b).



(**a**) (**b**)

Figure 5. Violin plot reporting the results in terms of (**a**) perplexity and (**b**) ROUGE-1 F1 score.

4.4. Supplementary Test Case: Large-Scale ER Schema Evaluation

To further validate the robustness of LLMs when applied to larger ER schemas, we designed an additional test case that involves a significantly extended conceptual model. The selected domain concerns university information management, encompassing key components such as students, courses, exams, and academic staff. The schema consists

of 84 total components, including 20 distinct entity types and 14 relationships of various types (both binary and ternary). This schema was selected for its complexity and representativeness of real-world database systems that require detailed documentation. The results are presented in Table 7, using GPT-4 as reference.

Table 7. Performance metrics of GPT-4 on large-scale ER schema.

Model	F1	PPL	ROUGE-1			ROUGE-2			ROUGE-L		
			P	R	F1	P	R	F1	P	R	F1
GPT-4	0.86	18.83	0.318	0.629	0.422	0.072	0.144	0.096	0.154	0.304	0.204

For this supplementary test case, GPT-4 achieved an F1 score of 0.86, which aligns with the average values observed in our main experiments (Section 4.1). This indicates that the approach maintains its effectiveness even when scaling to larger schemas, as the model was able to successfully identify all main elements of the schema.

However, we observed that the model occasionally provided simplified descriptions for more complex elements, such as composite attributes. The generated text was more concise and less redundant compared to the reference descriptions, resulting in some attributes or elements not being fully described. There are cases in which the model assumed certain properties as implicit, omitting their explicit mention while preserving the overall semantic meaning in a more condensed form. The high recall values (0.629 for ROUGE-1) indicate that the generated text still captures most of the key content, despite the compact style.

The perplexity score (PPL) of 18.83 demonstrates that the generated text maintained reasonable fluency and coherence, despite the complexity of the source schema. This is particularly important for documentation purposes, where the readability of the output text is crucial for end-users.

These results demonstrate that this LLM-based approach scales effectively to large, complex ER schemas while maintaining consistent performance across multiple evaluation metrics. The ability to capture all key elements while producing more concise descriptions suggests practical applicability in real-world database documentation scenarios where both comprehensiveness and clarity are essential.

5. Discussion and Conclusions

A key motivation for this work is the need for more readable and descriptive textual representations of database schemas. The generated text serves as a complement to ER data models, improving the clarity of the underlying structure and semantics. This is particularly useful for documentation and schema validation, where human-readable descriptions facilitate better communication among database designers and stakeholders.

The results of our exploratory analysis highlight that LLMs have strong potential in transforming ER schemas into human-readable text, offering a practical solution for automating database documentation. Unlike manually written documentation, which can become outdated or inconsistent with the evolving schema, LLM-generated descriptions remain aligned with the ER model, reflecting its most recent state.

We discussed the strengths and limitations of different LLMs in generating textual descriptions from ER conceptual schemas, as evaluated through F1 score, perplexity, and ROUGE metrics. The models accurately capture the main ER constructs, often producing well-structured text. However, some limitations emerge, particularly in the treatment of complex constructs (e.g., generalization). These elements are sometimes simplified or misrepresented, leading to inconsistencies in the generated descriptions. Enhancing the preprocessing of input schemas with specific hints improves the quality of the output. This

suggests that a hybrid approach, where human intervention helps structure the input data, could enhance the quality of the generated text. The output of older models such as GPT-3.5 and Llama 2 tend to be more general and less accurate. However, they are still valuable by relying on the additional prompt preprocessing to improve their descriptions.

Our study is primarily focused on benchmarking various LLMs to describe in natural language a database conceptual data model based on the ER schema. Future research can extend this work in several directions. First, applying LLMs to other conceptual modeling languages such as UML class diagrams would provide insights into their generalizability beyond ER models. Second, a comparison of other automatic architectures (e.g., Seq2Seq) with LLM-based solutions could be investigated. Lastly, our proposed methodology for non-relational databases, including document stores and graph databases, can be adapted to explore their effectiveness in more flexible schema environments.

Author Contributions: Conceptualization, A.A., S.C. and A.F.; methodology, A.A., S.C. and A.T.; software, A.T.; validation, A.A. and A.T.; formal analysis, S.C. and A.F.; investigation, A.A. and A.T.; resources, A.T. and A.F.; data curation, A.T. and S.C.; writing—original draft preparation, A.A., S.C. and A.T.; writing—review and editing, A.A. and S.C.; supervision, S.C. and A.F.; project administration, S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original data presented in this study are openly available in https://github.com/AndreaAvignone/ER2Text (accessed on 10 March 2025).

Acknowledgments: This study was partially carried out within the FAIR—Future Artificial Intelligence Research—and received funding from the European Union Next-GenerationEU (PNRR MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.3 D.D. 1555 11/10/2022, PE00000013), as well as partially supported by the SmartData@PoliTO center on Big Data and Data Science. This paper reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LLM	Large Language Model
ER	Entity-Relationship
SD	Standard Deviation
PPL	Perplexity Score

References

1. Atzeni, P.; Ceri, S.; Paraboschi, S.; Torlone, R. *Database Systems—Concepts, Languages and Architectures*; McGraw-Hill Book Company: New York, NY, USA, 1999.
2. Ramakrishnan, R.; Gehrke, J. *Database Management Systems*, 3rd ed.; McGraw-Hill: New York, NY, USA, 2003.
3. Silberschatz, A.; Korth, H.F.; Sudarshan, S. *Database System Concepts*, 6th ed.; McGraw-Hill: New York, NY, USA, 2010.
4. Kimball, R.; Ross, M. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*, 2nd ed.; Wiley: New York, NY, USA, 2002.
5. Yu, W.; Zhu, C.; Li, Z.; Hu, Z.; Wang, Q.; Ji, H.; Jiang, M. A Survey of Knowledge-enhanced Text Generation. *ACM Comput. Surv.* **2022**, *54*, 1–38. [CrossRef]
6. Chen, X.; Gao, C.; Chen, C.; Zhang, G.; Liu, Y. An Empirical Study on Challenges for LLM Application Developers. *ACM Trans. Softw. Eng. Methodol.* 2025, *just accepted*. [CrossRef]

7. Hou, X.; Zhao, Y.; Liu, Y.; Yang, Z.; Wang, K.; Li, L.; Luo, X.; Lo, D.; Grundy, J.; Wang, H. Large Language Models for Software Engineering: A Systematic Literature Review. *ACM Trans. Softw. Eng. Methodol.* **2024**, *33*, 1–79. [CrossRef]
8. Katsogiannis-Meimarakis, G.; Koutrika, G. A Survey on Deep Learning Approaches for Text-to-SQL. *VLDB J.* **2023**, *32*, 905–936. [CrossRef]
9. Guo, J.; Zhan, Z.; Gao, Y.; Xiao, Y.; Lou, J.G.; Liu, T.; Zhang, D. Towards Complex Text-to-SQL in Cross-Domain Database with Intermediate Representation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; Korhonen, A., Traum, D., Màrquez, L., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 4524–4535. [CrossRef]
10. Lee, J.S.; Hsiang, J. Patent claim generation by fine-tuning OpenAI GPT-2. *World Pat. Inf.* **2020**, *62*, 101983. [CrossRef]
11. Bień, M.; Gilski, M.; Maciejewska, M.; Taisner, W.; Wisniewski, D.; Lawrynowicz, A. RecipeNLG: A Cooking Recipes Dataset for Semi-Structured Text Generation. In Proceedings of the 13th International Conference on Natural Language Generation, Dublin, Ireland, 15–18 December 2020; Davis, B., Graham, Y., Kelleher, J., Sripada, Y., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 22–28. [CrossRef]
12. Avignone, A.; Fiori, A.; Chiusano, S.; Rizzo, G. Generation of Textual/Video Descriptions for Technological Products Based on Structured Data. In Proceedings of the 2023 IEEE 17th International Conference on Application of Information and Communication Technologies (AICT), Baku, Azerbaijan, 18–20 October 2023; pp. 1–7. [CrossRef]
13. Gong, H.; Sun, Y.; Feng, X.; Qin, B.; Bi, W.; Liu, X.; Liu, T. TableGPT: Few-shot Table-to-Text Generation with Table Structure Reconstruction and Content Matching. In Proceedings of the International Conference on Computational Linguistics, Barcelona, Spain, 8–13 December 2020; Scott, D., Bel, N., Zong, C., Eds.; International Committee on Computational Linguistics: New York, NY, USA, 2020; pp. 1978–1988. [CrossRef]
14. Ozkaya, I. Application of Large Language Models to Software Engineering Tasks: Opportunities, Risks, and Implications. *IEEE Softw.* **2023**, *40*, 4–8. [CrossRef]
15. Zheng, Z.; Ning, K.; Zhong, Q.; Chen, J.; Chen, W.; Guo, L.; Wang, W.; Wang, Y. Towards an Understanding of Large Language Models in Software Engineering tasks. *Empir. Softw. Engg.* **2024**, *30*. [CrossRef]
16. Marques, N.; Silva, R.R.; Bernardino, J. Using ChatGPT in Software Requirements Engineering: A Comprehensive Review. *Future Internet* **2024**, *16*, 180. [CrossRef]
17. Cai, R.; Xu, B.; Zhang, Z.; Yang, X.; Li, Z.; Liang, Z. An Encoder-Decoder Framework translating Natural Language to Database Queries. In Proceedings of the IJCAI'18: International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; AAAI Press: Washington, DC, USA, 2018; pp. 3977–3983.
18. Deng, N.; Chen, Y.; Zhang, Y. Recent Advances in Text-to-SQL: A Survey of What We Have and What We Expect. *arXiv* **2022**, abs/2208.10099.
19. Câmara, V.; Mendonca-Neto, R.; Silva, A.; Cordovil, L. A Large Language Model approach to SQL-to-Text Generation. In Proceedings of the 2024 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 5–8 January 2024; pp. 1–4. [CrossRef]
20. Zhang, X.; Yin, F.; Ma, G.; Ge, B.; Xiao, W. M-SQL: Multi-Task Representation Learning for Single-Table Text2sql Generation. *IEEE Access* **2020**, *8*, 43156–43167. [CrossRef]
21. Li, Q.; You, T.; Chen, J.; Zhang, Y.; Du, C. LI-EMRSQL: Linking Information Enhanced Text2SQL Parsing on Complex Electronic Medical Records. *IEEE Trans. Reliab.* **2024**, *73*, 1280–1290. [CrossRef]
22. Yu, T.; Li, Z.; Zhang, Z.; Zhang, R.; Radev, D. TypeSQL: Knowledge-Based Type-Aware Neural Text-to-SQL Generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), New Orleans, LA, USA*; Walker, M., Ji, H., Stent, A., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2018; pp. 588–594. [CrossRef]
23. Han, D.; Lim, S.; Roh, D.; Kim, S.; Kim, S.; Yi, M.Y. Leveraging LLM-Generated Schema Descriptions for Unanswerable Question Detection in Clinical Data. In Proceedings of the International Conference on Computational Linguistics, Abu Dhabi, United Arab Emirates, 19–24 January 2025; Rambow, O., Wanner, L., Apidianaki, M., Al-Khalifa, H., Eugenio, B.D., Schockaert, S., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2025; pp. 10594–10601.
24. Ferrari, A.; Abualhaija, S.; Arora, C. Model Generation with LLMs: From Requirements to UML Sequence Diagrams. In Proceedings of the 2024 IEEE 32nd International Requirements Engineering Conference Workshops (REW), Reykjavik, Iceland, 24–25 June 2024; pp. 291–300. [CrossRef]
25. Wang, B.; Wang, C.; Liang, P.; Li, B.; Zeng, C. How LLMs Aid in UML Modeling: An Exploratory Study with Novice Analysts. In Proceedings of the 2024 IEEE International Conference on Software Services Engineering (SSE), Shenzhen, China, 7–13 July 2024; pp. 249–257. [CrossRef]
26. Ardimento, P.; Bernardi, M.L.; Cimitile, M. Teaching UML using a RAG-based LLM. In Proceedings of the 2024 International Joint Conference on Neural Networks (IJCNN), Yokohama, Japan, 30 June–5 July 2024; pp. 1–8. [CrossRef]

27. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-shot Learners. In Proceedings of the NIPS '20: 34th International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 6–12 December 2020.
28. Sivarajkumar, S.; Kelley, M.; Samolyk-Mazzanti, A.; Visweswaran, S.; Wang, Y. An Empirical Evaluation of Prompting Strategies for Large Language Models in Zero-Shot Clinical Natural Language Processing. *arXiv* **2023**, abs/2309.08008.
29. Kojima, T.; Gu, S.S.; Reid, M.; Matsuo, Y.; Iwasawa, Y. Large Language Models are Zero-shot Reasoners. In Proceedings of the NIPS '22: 36th International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 28 November–9 December 2022.
30. Reynolds, L.; McDonell, K. Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm. In Proceedings of the CHI EA '21: Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems, New York, NY, USA, 8–13 May 2021. [CrossRef]
31. Min, S.; Lyu, X.; Holtzman, A.; Artetxe, M.; Lewis, M.; Hajishirzi, H.; Zettlemoyer, L. Rethinking the Role of Demonstrations: What Makes In-Context Learning Work? *arXiv* **2022**, abs/2202.12837.
32. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.H.; Le, Q.V.; Zhou, D. Chain-of-thought Prompting elicits reasoning in Large Language Models. In Proceedings of the NIPS '22: 36th International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 28 November–9 December 2022.
33. Zhang, Z.; Zhang, A.; Li, M.; Smola, A. Automatic Chain of Thought Prompting in Large Language Models. *arXiv* **2022**, arXiv:2210.03493.
34. Jelinek, F.; Mercer, R.L.; Bahl, L.R.; Baker, J.K. Perplexity—A Measure of the Difficulty of Speech Recognition Tasks. *J. Acoust. Soc. Am.* **1977**, *62*, S63. [CrossRef]
35. Colla, D.; Delsanto, M.; Agosto, M.; Vitiello, B.; Radicioni, D.P. Semantic Coherence Markers: The Contribution of Perplexity Metrics. *Artif. Intell. Med.* **2022**, *134*, 102393. [CrossRef]